TITLE OF THE INVENTION

Dynamic subject information generation in message services of distributed object systems

INVENTORS: Yueh-Shian T. Chi, Parris C.M. Hawkins, Charles Q. Huang

FIELD OF THE INVENTION

The present invention relates to message services in a distributed object system in which one or more message service servers interact with a number of publishers and subscribers. More specifically, the present invention relates to a system, method and medium for providing one or more message servers which communicate with one or more publishers configured to create and publish messages having changeable or selectable subject information, the published subject information being of potential interest to one or more of the subscribers.

BACKGROUND OF THE INVENTION

Distributed object systems are designed to increase the efficiency of computer program development by enabling object reuse and simplifying system maintenance through clear separation of functions. Each object in a distributed object system encapsulates the data for that object and the procedures or methods for operating on that data. Encapsulation means that the data for an object can be manipulated only by that object using the defined methods. These features of distributed object systems allow the objects to be reused and portable. Exemplary distributed object systems include: COM (Common Object Model), COM+, DCOM (Distributed Component Object Model) and CORBA (Common Object Request Broker Architecture).

One of the features of the distributed object system is a message service. A conventional message service system includes one or more publishers, subscribers and message servers. A publisher is a program (object or method) that makes calls that initiate sending messages that contain data, and a subscriber is another program (object or method) that receives the messages from a publisher. A subscriber indicates to (e.g., registers with) its message server that it wishes to receive messages from a publisher.

An exemplary conventional message service server is Message Queuing Services (MSMQ) developed by Microsoft may be used. MSMQ implements asynchronous message service by enabling applications (e.g., data providers) to send messages to other applications (e.g., data receivers). While the messages are being forwarded from senders to receivers, MSMQ keeps the messages in queues. The MSMQ queues may protect messages from being lost in transit and provide a place for receivers to look for messages when they are ready. MSMQ is configured to support IPX (Internet Packet eXchange) and TCP/IP (Transmission Control Protocol/Internet Protocol) networking protocols. In the distributed object system parlance, a publisher is a data provider (e.g., the method sending the message) and a subscriber is a data receiver (e.g., the method receiving the message).

The conventional distributed systems fall short when messages are to be exchanged between a large number of publishers and subscribers, because in such a case the conventional message service system is required to predefine the relation between the data providers and data receivers (e.g., certain types of messages are predefined to be received by certain subscribers). In

particular, the conventional system may provide adequate message services when all the relations are predefined and do not change. However, the conventional message system fails when the relations are to be dynamic. For example, assume a subset of the data providers are to send messages to one subset of the data receivers under one condition while the same subset of the data providers are required to send messages to another subset of the data receivers under another condition. Under such a scenario, the connections between data providers and data receivers are required to be updated dynamically (e.g., as the conditions change and/or as the messages are created).

SUMMARY OF THE INVENTION

Accordingly, embodiments of the present invention provides an improved system, method and medium of sending messages in a distributed object system. More specifically, embodiments of the present invention contemplate receiving a message that includes subject information that is generated based on one or more pre-selected portions as the message is generated. The message is then delivered based on the subject information. Embodiments of the present invention also contemplates a message delivery system in a client-server environment. The message delivery system may include a server configured to receive a message that includes subject information that is generated based on one or more pre-selected portions as the message is created and configured to forward the message based on the subject information.

BRIEF DESCRIPTION OF THE DRAWINGS

The detailed description of a preferred embodiment of the present invention showing various distinctive features over prior art message servers may be best understood when the detailed description is read in reference to the appended drawing in which:

FIG. 1 is a schematic representation of an exemplary distributed object system of the present invention;

FIG. 2 is a flow chart representation of exemplary interactions among servers, publishers, and subscribers of the present invention;

FIG. 3 is a flow chart representation of an exemplary packing step performed by a publisher of the present invention;

FIG. 4 is a flow chart representation of an exemplary unpacking step performed by a subscriber of the present invention;

FIG. 5 is a drawing depicting an exemplary graphical user interface configured to show design time subject information of a message created by embodiments contemplated by the present invention;

FIG. 6. is a drawing depicting an exemplary graphical user interface configured to show run-time subject information of a message created by embodiments contemplated by the present invention;

FIG. 7 is a block diagram of exemplary assembly lines implemented to use the message server of the present invention;

FIG. 8 is a flow chart representation of exemplary use of the message server of the present invention in the assembly lines depicted in FIG. 7;

FIG. 9 is a block diagram of an exemplary stock quote system implemented to use the message server of the present invention; and

FIG. 10 is a flow chart representation of exemplary use of the message server of the present invention in the stock quote system depicted in FIG. 9;

FIG. 11 is a block diagram representation of an exemplary embodiment of a computing system utilizable in aspects and environment of the present invention; and

FIG. 12 illustrates one example of a memory medium which may be used for storing a computer implemented process of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

In FIG. 1, for purposes of explanation and not limitation, embodiments of the present invention are described in a client-server environment of a distributed object system. (Other environments, such as peer-to-peer, are also contemplated.) Referring now to FIG.1, the exemplary client-server message service system 100 of the present invention includes a number of publishers 102, 104, subscribers 106, 108, and ES (Event Service) servers 110, 114. ES servers keep track of which subscribers desire to receive messages with certain subjects and direct the messages to those subscribers without requiring the subscribers to designate specific publishers. A registration may involve storing information relating to the registered subscribers (e.g., subject information about which the subscribers desire to receive messages and/or IP addresses of the subscribers to forward the messages). The letter "L" for publisher 104, "M" for ES server 114 and "N" for subscriber 108 represent different integer numbers to illustrate that embodiments of the present invention can include arbitrary numbers of publishers, ES servers and subscribers. It should be noted that publishers, subscribers and ES servers may be referring to objects and/or methods (i.e., the methods of those objects) depending upon the context in which they are referenced. In the parlance of the distributed object system, an object is a piece of code that owns features such as attributes and data, and provides services through methods (methods are also referred as operations or functions). In embodiments of the present invention, it is contemplated that the objects are coded using the C++ language, although it should be understood that other computer programming languages could also be used. In addition, it is also contemplated that objects can be implemented to be operational in COM (Common Object Model), COM+, DCOM (Distributed Component Object Model), CORBA (Common Object Request Broker Architecture) or other similar systems.

Publishers 102, 104, subscribers 106, 108 and ES servers 110, 114 typically reside in one or more computers. For example, the publishers may reside within a first group of computers, while the subscribers may reside within a second group of computers. Further, the ES servers may reside in a third group of computers. In an alternative embodiment, one or more publishers and subscribers may reside in one computer. In general, it should be understood that embodiments of the present invention envision that any number of computers may be utilized to include any combination of the publishers, subscribers and ES servers. The communication link among the publishers, subscribers and ES servers may include local area networks (LAN), wide area networks (WAN), the Internet (TCP/IP protocol), cable, optical, wireless or the like (or any combination thereof).

Each of messages to be serviced by the present invention preferably includes subject information (subject) and content information (content). The subject of a message characterizes and/or identifies it. The content is the substantive information of the message. In the parlance of the distributed object systems, the publishing of messages may be viewed as events.

A method of operation envisioned by embodiments of the present invention is described in conjunction with the flow chart of FIG. 2. Referring now to FIG. 2, embodiments of the present invention contemplate instantiating one or more publishers configured to forward messages to an ES server (step 202). And one or more subscribers register with one or more ES servers (step 204). In particular, for a subscriber to receive messages having a certain "subject", it is envisioned that the subscriber registers with one of ES servers.

When a subscriber is registered with an ES server, the ES server makes an entry into a registration database (e.g., a registration database 112 of FIG. 1 associated with ES server 110). The registration database is configured to store all or some pertinent information relating to the registered subscriber. For instance, the registration database may store information relating to a subject or subjects that the registered subscriber would like to receive from a publisher or publishers. The registration database may also store the destination addresses of the registered subscriber so that messages having the desired subject(s) may be transmitted thereto. Subscribers and publishers may add, delete and/or modify the registrations at any time by notifying the ES servers. The notified ES servers then update their corresponding registration databases.

In embodiments of the present invention, an ES server is allowed to be instantiated many times. In addition, each ES server notifies its registered subscribers to other instantiated ES servers. This feature allows a direct routing of published messages.

For instance, assume Publisher 1 is configured to fire (e.g., send) its messages to ES Server 1 and Subscriber N is registered with ES Server M. In embodiments of the present invention, ES Server M notifies ES Server 1 that Subscriber N is registered to receive messages having certain subject information. Accordingly, when a message having subject information identical to the subject information registered by Subscribe N is published by Publisher 1, the message is delivered directly from ES Server 1 to Subscriber N. The efficiency is gained by preventing the

message being routed from ES Server 1 to ES Server M before it can be delivered to Subscriber N.

It follows that embodiments of the present invention may be configured to increase the capacity of its messaging services (e.g., be linearly scalable). In other words, when more messaging services are required, more ES servers are instantiated to handle the increased capacity. Inembodiments of the present invention, there is no bottle neck that prevents the ES server to be linearly scalable.

A registration database may be stored and maintained in a memory bank at a remote location (117 of FIG. 1). In an alternative embodiment, the registration database may be stored in the memory of a computer in which the ES server resides. The content of such a registration database may be copied into one or more other ES servers. Accordingly, one or more ES servers may function as backups when the ES server that created the database is damaged or otherwise temporarily inoperable.

The next step as contemplated by embodiments of the present invention is for a publisher to create a message (e.g., an event) having a subject that is selectable as the message is generated (step 206). Step 206 is described in greater detail by referring to FIG. 3. Referring now to FIG. 3, in order to create a message, a publisher preferably creates GC (Generic Context) or a nested GC (step 302), creates subject and content of the message based on the GC (step 304), and packs the created message (step 306). A GC or a nested GC is a data structure with which the subject and content of messages are created, as described below.

First, in step 302, GC may include one or more context elements (e.g., possible subject elements). In an example contemplated herein, each context element may include a name, type, value and size thereof. Each context element could be of type integer, float, string, or a generic context (e.g., nested generic context), or an array of integers, an array float, an array of string, or an array of generic context. Table 1 below illustrates an exemplary nested GC. More specifically, the exemplary GC includes elements such as "LotNo," "Weight," "Loc," and "LotId" while it also includes arrays of GC elements designated as "GC1" and "GC2." Although not shown in Table 1, the size is also defined for each element.

Table 1

Element NO	Variable Name	Type	Value
1	LotNo	Int	125
2	Weight	Float	125.000
3	GC1	CGC_Context	Context1
4	Loc	BSTR	"MtnView"
5	LotId	Int	125
6	GC2	CGC_Context	Context2

The above described GC or nested GC forms exemplary content of a message to be created. In addition, embodiments of the present invention contemplate that for a message to be created, its "subject" is also generated. The subject can include two portions: a design time subject and a run-time subject. The design time subject (e.g., a fixed portion) is a portion of the subject that is specific to the publisher that created the message. Thus, each of the publishers has associated with it a design time subject, which becomes a part of the overall subject of a message generated by a publisher. In addition, as each message is created by a publisher, a run-time subject (e.g., a

changeable portion) is appended to the design time subject of the publisher. Unlike the design time subject, the run-time subject is selectable by a user (e.g., an operator or automated entity) . The subject, as it includes both the design time subject portion and the run-time subject portion, describes the format or the information relating to content (e.g., the values of each element).

As an example of the design time subject, FIG. 5 illustrates a graphical user interface which shows the design time subject of a publisher. More specifically, messages created by the illustrated publisher would always include "CSIM.CSIM_DISPREQUEST" 501 as their design time subject. It should be noted that the design time subject "CSIM.CSIM_DISPREQUEST" is only provided here as an example. Any sequence of characters uniquely identifiable is sufficient to meet the purpose of the present invention.

As an example of the run-time subject, FIG. 6 illustrates a graphical user interface which shows a list of all possible subject elements 601 that can be appended to the design time subject. As illustrated, each element includes name 602, data type 603, publish options 604, and position options 605. Accordingly, a user by using the graphical user interface depicted in FIG. 6 (or an automated mechanism by use of the graphical user interface or other facilities) may designate any combination of the elements to be published or not published as run-time subjects by making appropriate selections using the publish options 604, thereby producing configuration information of the run-time subject. The user is also allowed to include any combination of the elements to be part of the run-time subject or not by making appropriate selections using the position options 605. In the example selections illustrated in FIG. 6, the element "bstrDSNameSpace" element is selected to be published at position 1 of the run-time subject,

and the element "bstrDSName" is selected to be published at position 2 of the run-time subject. The user may enter strings to represent the above identified exemplary elements (e.g., a string "Assembly_line_1" for the "bstrDSNameSpace" element and a string "Etcher_1" for the "bstrDSName" element). Accordingly, when the fixed and selected information in the examples illustrated in FIGs. 5 and 6 are combined, the following subject information may be created:

CSIM.CSIM_DISPREQUEST.Assembly_line_1.Etcher_1

A user may also select the position of the "bstrDSNameSpace" element to be 2 and the position of the "bstrDSName" element to be 1. In this example, the subject information may be:

CSIM.CSIM_DISPREQUEST.Etcher_1.Assembly_line_1.

The above described feature of selecting run-time subject is preferably performed by a meta-data server which may be a part of an ES server. In particular, the meta-data server allows a user to select any combination of all available run-time subject elements and also allows the user to arrange the selected elements in any order using the position options. Once the user selects and arranges the run-time subject for a particular publisher, that information is stored in the meta-data server. Subsequently, when the publisher is creating a message, based on its design time subject, the publisher retrieves the user selection and arrangement information from the meta-data server. The publisher then creates the message with subject information, the run-time subject of which is generated according to the retrieved information. Embodiments of the present invention contemplate that the user selection and arrangement information can be retrieved from the meta-data server and stored in the publisher's cache memory. In this scenario,

unless the information in the cache memory is lost, it is envisioned that the meta-data server is not accessed again to retrieve the user selection and arrangement information.

Although not shown in FIG. 6, in this example the value of each element is also produced when the messages are created. The values represent data of each element (i.e., at least part of the content of the message). Accordingly, the GC context is used to form both the content format (e.g., information) and subject of the messages to be created.

Referring back to FIG. 3, a message that includes content and subject as described above is created as the result of step 304. The created message is then packaged into a stream of bits (step 306). In the parlance of the distributed object system, this step is called marshalling the message. One message at a time may be created, or one or more messages may be created simultaneously or in a sequence. The created messages are then fired (e.g., sent) to the ES server which is configured to route the messages for the publisher (step 208 of FIG. 2).

Referring back to FIG. 2, upon receiving the messages from the publisher, the ES server then searches its corresponding registration database to identify any registered subscribers that have registered to receive messages having the subject of the messages from the publisher. If there are any, the messages are then forwarded to those identified subscribers (step 210 of FIG. 2).

Referring back to FIG. 1, and as an example of the previously-described concepts, assume Subscriber 1 registers with ES Server 1 to receive messages having the subject of "Design_Time_Subject_1.run_time_subject_N." In this scenario, ES Server 1 also notifies one

or more ES servers (e.g., ES Server M) to update their corresponding registration databases regarding subscriber 1. Subsequently, if Publisher L fires a message to ES Server M having a subject of "Design_Time_Subject_1.run_time_subject_N," then the message is preferably delivered to Subscriber 1 directly from ES Server M.

Embodiments of the present invention contemplate that a guarantee message delivery (GMD) mechanism can be provided. GMD can be used when hardware (e.g., computers) and network connections may be unstable.

In operation of the GMD feature, the ES servers keep messages for the subscriber when the subscriber is off-line. At the same time, a system-wide timeout period is set for each message. If a timeout happens before the subscriber retrieves its message, the message is preferably re-routed to an ES server where a predefined action may be taken. For instance, if a subscriber designated to receive a message is off-line for longer than a predetermined period (e.g., 5 minutes, a day, etc.), the ES server preferably sends a notice (e.g., a pager, an e-mail message, etc.) to an operator to take appropriate action.

Referring to FIG. 4, upon receiving a message, a subscriber unpacks the message which is in a binary stream format (step 402). In the parlance of the distributed object system, this step is referred as unmarshaling. When the message is unpacked, it is converted into the GC context format which was described above and exemplified in Table 1.

The messaging service of the present invention is described below in a pair of practical exemplary applications. Although only two applications are illustrated below, other applications in which messaging service utilizing publisher/subscriber configuration (e.g., a weather monitoring system, medical monitoring system or the like) are also contemplated within this invention and may be obvious variations of the present the exemplary applications described below.

The first exemplary illustration is described in conjunction with FIGs 7 and 8. Referring first to FIG. 7, a microelectronic manufacturing system 701 configured to use in embodiments of the present invention includes a number of assembly lines 702, 710. Each assembly line includes manufacturing devices such as a number of etchers 703, 705, 711, 713 and layer depositors 707, 709, 715, 717. The manufacturing system also includes one or more controllers 719, 721. The letter "L" for etcher 705 in assembly line 1, "M" for layer depositor 709 in assembly line 1, "N" for etcher 713 in assembly line Q, "O" for layer depositors 717 in assembly line Q, "P" for controller 721 and "Q" for assembly line 710 represent different integer numbers to illustrate that embodiments of the present invention anticipate the utilization of any number of the designated items.

For instance, an etcher is a manufacturing apparatus configured to etch a layer or layers of a substrate during manufacture of an electronic device. Similarly a layer depositor is an apparatus configured to deposit a layer or layers on a substrate during manufacture of an electronic device. Preferably, assembly line devices (e.g., etchers, depositors) and controllers include a computer or computer-like device that includes a processor, a read-only memory device and a random access

memory. Each of the assembly line devices and controller are also preferably configured to include at least one subscriber and one publisher operating in their respective computer or computer-like devices. In an alternative embodiment, one or more assembly line devices may include only a subscriber or a publisher operating in their respective computer or computer-like devices. In another alternative embodiment, one or more assembly line device may include no subscriber or publisher operating in their respective computer or computer-like devices. In yet another embodiment, one or more assembly line device may not include any computer or computer like devices.

One or more ES servers can reside in one or more server computers (not shown in FIG. 7). In an alternative embodiment, some or all of the ES servers may reside in any of the assembly line devices and/or controllers. The communication links among the assembly line devices, controllers and server computers, may include a local area network (LAN), wide area network (WAN), the Internet (TCP/IP protocol), cable, optical, wireless or the like (or any combination of them).

Embodiments of the present invention contemplate that, the manufacturing system depicted in FIG. 7 may be located in one facility. In an alternative embodiment, one of the assembly lines may be located in one facility while another one of the assembly lines may be located in a remotely located facility. In addition, the controllers may be located in one facility separated from the facility(ies) in which the assembly lines are located.

Now referring to FIG. 8, publishers are preferably instantiated and reside in one or more controllers and in the assembly line devices (steps 81, 83). Subscribers, which may reside in one or more controllers and in the assembly line devices, can also be registered with the one or more ES servers (steps 802, 804). Once the registration is completed (step 805), the message service may be activated. It should be noted, however, that embodiments of the present invention contemplate that subscribers and publishers may add, delete and/or modify the registrations at any time.

Subsequently, one or more subscribers located among the assembly line devices may create one or more messages (step 806). Such messages may include content information relating to, for example, the operating temperature of the devices, status of processes (e.g., etching or deposition) and/or maintenance information. Depending upon the character of the content information, the run-time subject is generated, which is appended to corresponding design time subjects.

Table 2

Serial #	Metadataname	Data type	pub_flag	sub_pos
1.	EntityNamespace	Text	1	1
2.	EntityID	Text	1	2
...	...	...	...	...
5.	Quantity	Integer	0	-1

Table 2 above illustrates a detailed example of information relating to generating a run-time subject. In particular, the exemplary run-time subject may include five possible elements: "EntityNamespace," "EntityID" ... "Quantity." Each element is defined by its name (i.e., metadata name), data type, and a publish flag and subject position definition. In this example, assume that the "EntityNamespace" element is defined as "Assembly line Q," and the EntityID element is defined as "Etcher_1." The "pub_flag" specifies whether particular element is to be published as part of the run-time subject: "1" designates that the element is to be published, "0" designates that the element is not to be published , and "-1" designates that the element is not to be included as a part of the runtime subject. If the element is to be published, then the "sub_pos" element specifies where the element is to be placed in the run-time subject. Assuming the design time subject is a string "Design_Time_Subject_1," then the subject specified in the above example may appear as the following:

Design_Time_Subject_1.Assembly_Line_Q.Etcher_1

A user, by means of a graphical user interface similar to the one depicted in FIG. 6, is allowed to specify which of the elements are to be published and, if to be published, the location in which the elements are to be placed in the run-time subject. When messages are created with the appropriate content and subject, the messages are fired to the one or more ES servers configured to receive the messages.

Upon receiving the messages, the one or more ES servers refer to their corresponding registration databases to identify which one or more of registered subscribers registered interest

in receiving messages having subjects that include the subjects of the received messages. The messages are forward to the identified subscribers (step 807).

In continuing with the above example described in connection with Table 2, assume two subscribers registered to receive messages having the following subjects:

First subscriber: Design_Time_Subject.Assembly_Line_Q (option field)

Second subscriber: Design_Time_Subject.Assembly_Line_Q.Etcher_1 (option field)

Depending upon a selection made in the option field, messages are delivered to various registered subscribers. The option field is configured to provide flexibility in matching subject information registered by the registered subscribers and the subject information provided by the published messages.

Examples of the use of the option field as contemplated by embodiments of the present invention will now be described. First, for the sake of this example, assume that the existence of a given uniquely identifiable character (e.g., ">"), means that any message that includes the subject elements previous to the character or any other elements appended thereto would be delivered to a subscriber registered with such an option. Thus, for example, assume further that the registered subject of the first subscriber is as follows:

Design_Time_Subject.Assembly_Line_Q.>

In this example, any message that includes "Design_Time_Subject.Assembly_Line_Q" or any other elements appended thereto (e.g., Design_Time_Subject.Assembly_Line_Q.Etcher_1, Design_Time_Subject.Assembly_Line_Q.Etcher_2.Quantiy_1, etc.) as its subject would be delivered to the first subscriber.

It also follows that if the second subscriber has registered its subject as "Design_Time_Subject.Assembly_Line_Q.Etcher_1.>", then any message that includes "Design_Time_Subject.Assembly_Line_Q.Etcher_1" or any other elements appended thereto as their subject would be delivered to the second subscriber.

In another example, assume that the existence of another uniquely identifiable character ("*"), means that any message that includes the subject elements previous to the character or one element appended thereto would be delivered to a subscriber registered with such an option. Thus, for example, assume further that the registered subject of the first subscriber is as follows:

Design_Time_Subject.Assembly_Line_Q.*

In this example, any message that includes "Design_Time_Subject.Assembly_Line_Q" or one other element appended thereto (e.g., Design_Time_Subject.Assembly_Line_Q.Etcher_1, Design_Time_Subject.Assembly_Line_Q.Etcher_2, etc.) as its subject would be delivered to the first subscriber. The second subscriber functions similarly.

The above relationship between subjects of a message to be delivered and the subscribers is also applicable without regard to the number of elements specified. It should be noted that the above provided examples of option field characters (e.g., "*" and ">") are provided only as sample examples. Other conventions available in the art are contemplated by embodiments of the present invention.

In this example, it is envisioned that, the subscribers residing in the controllers register with the ES servers to receive messages from the publishers residing in the assembly line devices. Alternatively and/or in addition, one or more subscribers located in one or more assembly line devices may register to receive messages published by one or more publishers residing in other assembly line devices. In any event, and referring back to FIG. 8, when the messages are received by the subscribers, they are unpacked, and the contents of the unpacked messages are then retrieved and processed by the controllers (or assembly line devices) (step 808).

For instance, assume a controller is configured to monitor temperatures of one or more etchers to prevent them from overheating, then such a controller may receive messages that include temperature information created by the one or more etchers. When a message received from one of the etchers indicate an overheating condition, the controller may decide to send a message (e.g., create a command, as indicated by step 808) to the etcher or to every manufacturing device on that assembly line to shut down.

After determining that one or more messages are desired to be forwarded to one or more assembly line devices, (which, in this example, are envisioned to contain certain commands) messages with corresponding contents and subjects are created (step 809). The created messages are then fired (step 810) to the ES servers which forwards messages to appropriate subscribers based on corresponding registration databases (step 811). When the message is delivered to one or more subscribers (and/or controllers) they are unpacked (step 812). The unpacked message (and any commands therein) is processed to take further action, if any.

Continuing with the above example of an overheated etcher, when the etcher receives the message which contains a command to shut down, the etcher preferably shuts itself down. If messages are sent to each of the assembly line devices in that assembly line, then the whole assembly line may shut itself down.

It should be noted that sending and receiving messages among different assembly line devices and controllers are not required to occur in the sequence describe above. More specifically, one or more of the assembly line devices may be receiving messages from one or more controllers while one or more of the assembly line devices may be creating and firing a message to one or more controllers. In addition, one or more assembly line devices and controllers may be receiving, creating and firing messages simultaneously - a multithreaded processing feature ofembodiments of the present invention.

The second exemplary illustration is now described with regard to FIGs 9 and 10. Referring to FIGs. 9 and 10, there is shown a stock market price quote example 901 implemented using the features of the present invention. The quote example includes a stock quote publisher 902, a computer or a computer-like device, configured to monitor stock prices of companies listed in a stock market (e.g., DOW, NASDAQ, S&P 500, etc.). The stock quote system 901 further includes one or more ES servers 903 and a number of subscribers 904, 905, 906. The subscribers register with the one or more ES servers to receive stock quotes of various companies.

For instance, subscriber 1 registers to receive stock price quotes of companies named from AA to BB and subscriber 2 registers to receive stock price quotes of companies named from CC to DD.

Subsequently, the publisher creates messages containing stock quotes of the stock trading companies. Then, the messages are fired to the ES servers. Upon receiving the message, ES servers then forward the message to appropriate subscribers. For instance, if the message includes a stock quote of company AB, then it would be sent to Subscriber 1, and so on.

FIG. 11 illustrates a block diagram of one example of the internal hardware of a computer system 1111 that includes one or more of publishers, subscribers and ES server of FIG. 1. A bus 1156 serves as the main information highway interconnecting the other components of system 1111. CPU 1158 is the central processing unit of the system, performing calculations and logic operations required to execute the processes of the present invention as well as other programs. Read only memory (ROM) 1160 and random access memory (RAM) 1162 constitute the main memory of the system. Disk controller 1164 interfaces one or more disk drives to the system bus 1156. These disk drives are, for example, floppy disk drives 1170, or CD ROM or DVD (digital video disks) drives 1166, or internal or external hard drives 1168. These various disk drives and disk controllers are optional devices.

A display interface 1172 interfaces display 1148 and permits information from the bus 1156 to be displayed on display 1148. Display 1148 is also an optional accessory. For example, display 1148 could be substituted or omitted. Display 1148 may be used in displaying graphical user interface as shown in FIGs. 5 and 6. Communications with external devices such as the other

components of the system described above, occur utilizing, for example, communication port 1174. Optical fibers and/or electrical cables and/or conductors and/or optical communication (e.g., infrared, and the like) and/or wireless communication (e.g., radio frequency (RF), and the like) can be used as the transport medium between the external devices and communication port 1174. Peripheral interface 1154 interfaces the keyboard 1150 and mouse 1152, permitting input data to be transmitted to bus 1156. In addition to these components, system 1111 also optionally includes an infrared transmitter and/or infrared receiver. Infrared transmitters are optionally utilized when the computer system is used in conjunction with one or more of the processing components/stations that transmits/receives data via infrared signal transmission. Instead of utilizing an infrared transmitter or infrared receiver, the computer system may also optionally use a low power radio transmitter 1180 and/or a low power radio receiver 1182. The low power radio transmitter transmits the signal for reception by components of the production process, and receives signals from the components via the low power radio receiver. The low power radio transmitter and/or receiver are standard devices in industry.

Although system 1111 in Fig. 11 is illustrated having a single processor, a single hard disk drive and a single local memory, the system 1111 is optionally suitably equipped with any multitude or combination of processors or storage devices. For example, system 1111 may be replaced by, or combined with, any suitable processing system operative in accordance with the principles ofembodiments of the present invention, including sophisticated calculators, and hand-held, laptop/notebook, mini, mainframe and super computers, as well as processing system network combinations of the same.

FIG. 12 is an illustration of an exemplary computer readable memory medium 1284 utilizable for storing computer readable code or instructions. As one example, medium 1284 may be used with disk drives illustrated in FIG. 11. Typically, memory media such as floppy disks, or a CD ROM, or a digital video disk will contain, for example, a multi-byte locale for a single byte language and the program information for controlling the above system to enable the computer to perform the functions described herein. Alternatively, ROM 1160 and/or RAM 1162 illustrated in FIG. 11 can also be used to store the program information that is used to instruct the central processing unit 1158 to perform the operations associated with the instant processes. Other examples of suitable computer readable media for storing information include magnetic, electronic, or optical (including holographic) storage, some combination thereof, etc.

In general, it should be emphasized that the various components of embodiments of the present invention can be implemented in hardware, software or a combination thereof. In such embodiments, the various components and steps would be implemented in hardware and/or software to perform the functions ofembodiments of the present invention. Any presently available or future developed computer software language and/or hardware components can be employed in such embodiments of the present invention. For example, at least some of the functionality mentioned above could be implemented using Visual Basic, C, C++, or any assembly language appropriate in view of the processor(s) being used. It could also be written in an interpretive environment such as Java and transported to multiple destinations to various users.

The many features and advantages ofembodiments of the present invention are apparent from the detailed specification, and thus, it is intended by the appended claims to cover all such features and advantages of the invention which fall within the true spirit and scope of the invention. Further, since numerous modifications and variations will readily occur to those skilled in the art, it is not desired to limit the invention to the exact construction and operation illustrated and described, and accordingly, all suitable modifications and equivalents may be resorted to, falling within the scope of the invention.